TWO-STAGE METHOD FOR SPECIFIC AUDIO RETRIEVAL

Wei-Qiang Zhang, Jia Liu

Department of Electronic Engineering, Tsinghua University, Beijing 100084, China
weiq.zhang@gmail.com, liuj@tsinghua.edu.cn

ABSTRACT

Specific audio retrieval, also referred as similarity-based audio retrieval, means to detect and locate a given query audio segment in a long stored audio signal. In this paper, we proposed a two-stage method for specific audio retrieval. In the first stage, the histogram pruning algorithm is used for coarse detection. In the second stage, the partial distance technique is used for fine verification and localization. Experimental results show that the two-stage coarse-to-fine method offers fast search speed and improves the robustness to additive noise and compression encoding.

Index Terms— Audio retrieval, histogram pruning, two-stage method

1. INTRODUCTION

With the rapid development of multimedia technologies, people now can share more types of media for entertainment, education, business or other purposes. Meanwhile, there are increasing interests in how to organize and retrieve the non text-based data [1]. This brings forth the multimedia retrieval.

In the field of multimedia retrieval, audio retrieval is always one of the most attractive but difficult problems [2]. Audio retrieval can be categorized into the content-based retrieval and similarity-based retrieval (also referred as specific audio retrieval). The former usually employs high-level information and includes audio indexing, keyword spotting, music retrieval and so on [3]. The latter deals with detecting a known query signal in a long stored signal (or stream) [4].

Specific audio retrieval, perhaps does not have so many applications as content-based audio retrieval. But for some special applications, such as advertisement monitoring, copyright management, etc., it can provide an efficient and effective solution. In addition, as a basic technology [5], specific audio retrieval may clue approaches for handling other multimedia information.

So far, the most widely used method for specific audio retrieval may be the histogram pruning algorithm [4–6]. It is an efficient algorithm, but it does not reflect the time order of feature vectors. Although sub-window method is introduced [5], this problem is only partly solved.

In this paper, we propose a two-stage method for specific audio retrieval, which can make full use of time order information. The rest of this paper is organized as follows. In section 2, we introduce the histogram pruning algorithm, which is used as the first stage in our method for coarse detection. Section 3 addresses our considerations on feature extraction. In section 4, we give the framework of the two-stage method and implementation the second stage. Detailed experiments and performance comparison are presented in section 5. Finally, section 6 gives conclusions.

2. OVERVIEW OF THE HISTOGRAM PRUNING ALGORITHM

A straightforward method for similarity-based search is employing the correlation coefficients between the query signal and the stored signal. This method, however, is a brute force search method and has two obvious disadvantages: time-consuming and sensitive to some audio coding. In fact, the correlation work can be done in the feature domain, i.e., match the similarity between the features of the query signal and the stored signal. But it still costs considerable time especially for long-running stored signals.

In order to solve this problem, Kashino *et al.* proposed a histogram pruning algorithm [4, 5], whose block diagram is shown in Fig. 1. The key accelerating technique is an effective pruning by using feature histograms. Suppose the feature histograms of the query signal and the stored signal segment (which has equal length with the query signal) are $\boldsymbol{h}^Q = (h_1^Q, h_2^Q, \cdots, h_B^Q)$ and $\boldsymbol{h}^S = (h_1^S, h_2^S, \cdots, h_B^S)$, respectively, where B is the number of histogram bins and h_i is the frequency counts in i-th bin. The similarity between $\boldsymbol{h}^Q$ and $\boldsymbol{h}^S$ can be defined as

$$S(\boldsymbol{h}^Q, \boldsymbol{h}^S) = \sum_{i=1}^{B} \min(h_i^Q, h_i^S). \tag{1}$$

If we have known the similarity between the query signal and the stored signal segment located at l_1 (which is the frame index), the upper bound of the similarity at l_2 can be predicted as

This project is partly supported by the National Natural Science Foundation of China (No. 60572083).

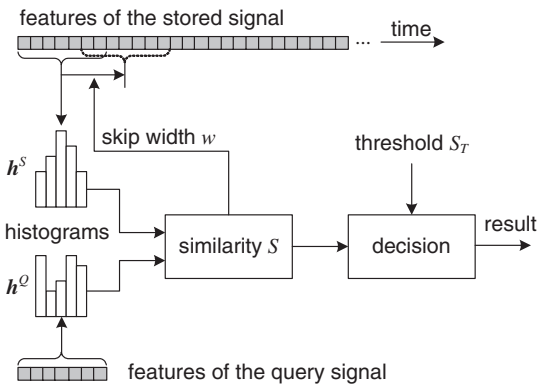

Fig. 1. Block diagram of the histogram pruning algorithm.

$$S^u(\boldsymbol{h}^Q, \boldsymbol{h}^S(l_2)) = S(\boldsymbol{h}^Q, \boldsymbol{h}^S(l_1)) + (l_2 - l_1). \quad (2)$$

According this upper bound and the detection threshold S_T, we can skip w frames from location l_1 without losing any precision.

$$w = \begin{cases} S_T - S(\boldsymbol{h}^Q, \boldsymbol{h}^S(l_1)) + 1, & \text{if } S(\boldsymbol{h}^Q, \boldsymbol{h}^S(l_1)) < S_T, \\ 1, & \text{otherwise.} \end{cases} \quad (3)$$

Using the skip width, the matching calculations can reduced by hundreds of times [5].

For one-dimensional features, we can easily count the number of occurrences of each feature to obtain the feature histogram. For higher-dimensional features, we can first map them to scalar codes by using vector quantization (VQ) algorithm [7], and then count the number of occurrences of each quantized code.

3. FEATURE EXTRACTION

In the audio retrieval field, various types of features have been proposed, such as the zero-crossing rate (ZCR) [4], normalized short-time power spectrum [5], normalized delta short-time power spectrum [8], dominant features obtained via eigen-decomposition [9], multiple feature vectors including ZCR, spectral centroid, spectral roll-off, and spectral flux [6].

For the specific audio retrieval task, the features should not only provide sufficient discriminative information, but also have low computational complexity. In this paper, we use the cepstral coefficients. The audio signal is first windowed with Hanning window, and then the discrete short-time Fourier transform (STFT), $X(\omega_k)$, is computed. After that, the energies of n-th sub-band can be given by

$$e(n) = \frac{1}{U_n - L_n + 1} \sum_{k=L_n}^{U_n} |X(\omega_k)|^2, \quad (4)$$

where L_n and U_n denote the lower and upper frequency indices of each sub-band.

In Reference [5], sub-band energies $\{e(n), n = 0, 1, \ldots, N-1\}$ are normalized by its maximum elements. Here, we first perform a logarithm operation and get $\log[1 + e(n)]$. (Note that usually $e(n) >> 1$, so $\log[1 + e(n)] \approx \log[e(n)]$. Adding 1 can avoid the log-energy $\rightarrow -\infty$ when $e(n) \rightarrow 0$.) It can compress the dynamic range and reduce the quantization bits, just like the μ-law transformation.

If we normalize the log-energies, it will be sensitive to the amplitude scale which may be introduced in some audio coding. We can find that if $e(n)$ scales to $\alpha e(n)$, the log-energy will be

$$\log[1 + \alpha e(n)] \approx \log[\alpha e(n)] \approx \log(\alpha) + \log[1 + e(n)]. \quad (5)$$

This means that amplitude scaling brings a offset in log-energies. We can use the discrete cosine transform (DCT),

$$c(m) = \sum_{n=0}^{N-1} \log[1 + e(n)] \cos \frac{2\pi(n + 1/2)m}{N}, \quad (6)$$

to get rid of it by discarding $c(0)$. Another advantage of the DCT is that it is close to the Karhunen-Loeve transform and thus it tends to decorrelate the original log-energies [10]. We use $\{c(m), m = 1, 2, \ldots, M, M \leq N\}$ as the feature, which is in fact a type of cepstral coefficients.

4. TWO-STAGE METHOD

The histogram pruning algorithm gains fast search speed at the cost of losing time order information of the query signal. For example, if the VQ code series of the query signal and the stored signal are $\{a, b, a, c, d\}$ and $\{b, a, d, a, c\}$, respectively, their histograms will be identical and $S(\boldsymbol{h}^Q, \boldsymbol{h}^S)$ will achieve maximum. In the practice applications, this may bring *false acceptance*. On the other hand, if the codebook used in VQ can not describe the feature vectors well, i.e., when VQ has much distortion, using the histogram pruning algorithm solely will not achieve good results, even when the sub-window method is also employed.

In fact, we can add a second stage after the histogram pruning to refine the results, as shown in Fig. 2. In the second stage, we use distance of the features as the refining criterion, which can make full use of the temporal information. Considering the computational complexity, we can select the absolute distance:

$$d(l) = \sum_{p=1}^{P} \sum_{m=1}^{M} |c_p^Q(m) - c_{l+p}^S(m)|, \quad (7)$$

where P is the number of frames of the query signal, and M is the dimension of the feature. $\{c_p^Q(m), m = 1, 2, \ldots, M\}$ and $\{c_p^S(m), m = 1, 2, \ldots, M\}$ are the features of p-th frame of

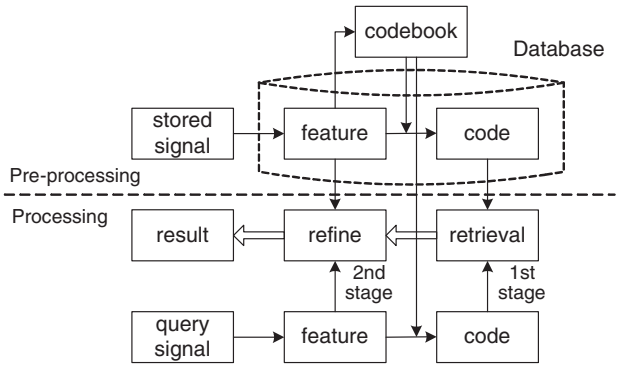

Fig. 2. Block diagram of the proposed two-stage method.

the query signals and stored signal, respectively, and l denotes the location of the stored signal segment.

Suppose the first stage output a coarse location l_c. In the second stage, we find a location l_f which gives the minimum distance in the interval $\mathscr{L} = [l_c - L, l_c + L]$. (Here, we simply assume $l_c - L$ and $l_c + L$ does not exceed the boundaries of the stored signal.)

$$l_f = \arg\min_{l \in \mathscr{L}} d(l). \tag{8}$$

Compare $d(l_f)$ with the distance threshold d_T, we can determine accept or reject it.

In order to improve the search speed, we borrow a *partial distance* technique, which was originally proposed by Chen *et al.* for fast vector quantization [11]. Its basic idea is that during the calculation of the distance, if the partial distance exceeds the previous minimum distance, this location is rejected without completing the distance calculation.

Because the location l_f is more likely located at l_c, this means that the minimum distance is more likely achieved when $l = l_c$. In the partial distance searching, the earlier the minimum distance is achieved, the more easily the partial distance is rejected. So we search from l_c to the two sides, which can further reduce the computational burden. The refine stage can be summarized as follows.

1. Calculate the distance at location l_c and set $d_{\min} := d(l_c)$, $l_{\min} := l_c$.
2. For $i = 1, 2, \cdots, L$:
   (a) Set $l := l_c + i$, and perform a partial distance search. If the calculation is not broken, it means $d(l) < d_{\min}$, then $d_{\min} := d(l)$, $l_{\min} := l$.
   (b) Set $l := l_c - i$, and perform a partial distance search. If the calculation is not broken, it means $d(l) < d_{\min}$, then $d_{\min} := d(l)$, $l_{\min} := l$.
3. If $d_{\min} \le d_T$, then accept it; else, reject it.

Table 1. Pre-processing time for 1-h signal

Step	CPU Time
Feature Extraction	3.60 s
Vector Quantization	11.27 s

Table 2. Search time for 10-h stored signal

Stage	Query Signal Duration		
	2 s	5 s	10 s
First Stage	0.0733 s	0.0741 s	0.0798 s
Second Stage	0.0028 s	0.0029 s	0.0040 s

5. EXPERIMENTAL RESULTS

In this section, we present some typical simulation results to demonstrate the performance of the proposed algorithm. In the feature extraction, the frame length was 0.016 s, and the frame step was 0.008 s. The frequency band of 0-4 kHz were equally divided into 16 sub-bands ($N = 16$), and 8 cepstral coefficients were chosen as the feature ($M = 8$). In VQ, the codebook size was 1024, which is equal to the number of the histogram bins ($B = 1024$). All the simulations were done on a microcomputer (AMD Sempron, 1.6 GHz).

5.1. Search speed

In this experiment, we investigated the search speed of the two-stage method. The stored signal was 10-h news broadcast (whose sampling rate was 8 kHz). The query signals were 100 randomly chosen segments from the stored signal and each query signal occurred once in the stored signal.

The central processing unit (CPU) time needed for pre-processing 1-h signal is listed in Table 1. We can see that the pre-processing totally costs approximately 0.4% of the signal duration.

Besides the stored signal duration, the searching time mainly depends on the detection threshold. We set $S_T = 0.6P$, $d_T = 20.0P$, where P is the frame number of the query signal. In the second stage, the search scope was $P/5$ frames ($L = P/5$). Using these parameters, there was neither false acceptance nor false rejection in the experiment. The average search time for one query is listed in Table 2. We can see that the time used in the second stage is about 4% of that in the first stage.

5.2. Search accuracy

We performed two experiments to evaluate the search accuracy. In the experiments, the stored signal was 70 MPEG Layer3 (MP3) songs (about 5 hours) whose bit rates are 128 kbps. The query signals were 1000 randomly chosen 2-s segments from the stored signals. We use the equal error rate (EER), which is achieved when the false acceptance

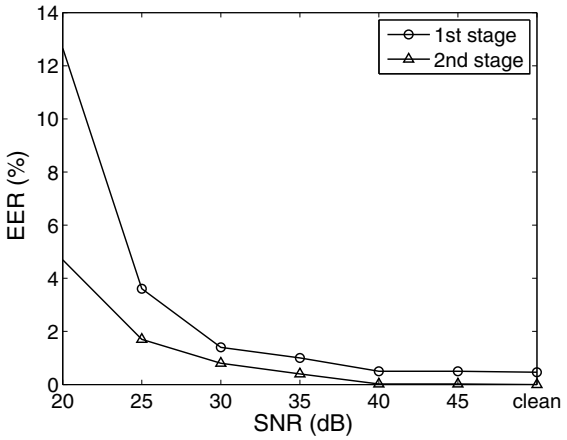

Fig. 3. Search accuracy (EER versus SNR).

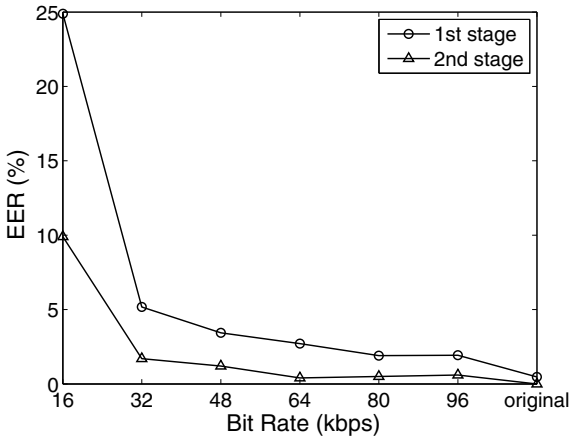

Fig. 4. Search accuracy (EER versus bit rate).

rate equals the false rejection rate by adjusting the detection threshold, as the evaluation criterion.

In the first experiment, we added white Gaussian noise to the query signals with different signal-to-noise ratios (SNRs). The EERs are plotted in Fig. 3. We can observe that for each stage, the EERs are both decreasing with the increasing of SNRs. After the second stage refining, the EERs are relatively reduced about 50%.

In the second experiment, we compressed the query signals with different bit rates. The EERs are illustrated in Fig. 4. Similar with the above experiment, the second stage also outperform the first stage.

Note that through other experiments, we found that the performance will be much better if the signal is speech or the query signal has longer duration. This is probably a consequence of that the features we used are more suitable to depict speech. These two simulations only give a conservative estimate of the proposed method.

6. CONCLUSIONS

This paper has proposed a two-stage method for specific audio retrieval. The search process is divided into two stages. The first stage is a coarse retrieval process, which utilize the histogram pruning algorithm. The second stage is a fine search and verification process, which employs partial distance technique. The experiments showed that via the second stage refine, the search accuracy can be obviously improved while the search time only increase about 4%.

In addition, although we only discussed the specific audio retrieval in this paper, the proposed method can be applied to specific video retrieval. Further research needs to be done.

7. REFERENCES

[1] Y. Wang, Z. Liu, and J.-C. Huang, "Multimedia content analysis-using both audio and visual clues," *IEEE Signal Processing Magazine*, vol. 17, no. 6, pp. 12–36, Nov. 2000.

[2] J. Foote, "An overview of audio information retrieval," *Multimedia Systems*, vol. 7, no. 1, pp. 2–10, Jan. 1999.

[3] J.H.L. Hansen, R. Huang, B. Zhou, et al., "Speechfind: Advances in spoken document retrieval for a national gallery of the spoken word," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 5, pp. 712–730, Sept. 2005.

[4] G. Smith, H. Murase, and K. Kashino, "Quick audio retrieval using active search," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'98)*, Seattle, May 1998, pp. 3777–3780.

[5] K. Kashino, T. Kurozumi, and H. Murase, "A quick search method for audio and video signals based on histogram pruning," *IEEE Transactions on Multimedia*, vol. 5, no. 3, pp. 348–357, Sept. 2003.

[6] K.-M. Kim, S.-Y. Kim, J.-K. Jeon, et al., "Quick audio retrieval using multiple feature vectors," *IEEE Transactions on Consumer Electronics*, vol. 52, no. 1, pp. 200–205, Feb. 2006.

[7] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*, Prentice Hall PTR, Upper Saddle River, NJ, 1993.

[8] W. Liang, S. Zhang, and B. Xu, "A histogram algorithm for fast audio retrieval," in *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR 2005)*, London, September 2005, pp. 586–589.

[9] J. Gu, L. Lu, R. Cai, et al., "Dominant feature vectors based audio similarity measure," *Lecture Notes in Computer Science*, vol. 3332, pp. 890–897, 2004.

[10] T. F. Quatieri, *Discrete-Time Speech Signal Processing: Principles and Practice*, Prentice Hall PTR, Upper Saddle River, NJ, 2002.

[11] D.-Y. Cheng, A. Gersho, B. Ramamurthi, et al., "Fast search algorithms for vector quantization and pattern matching," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'84)*, San Diego, Mar. 1984, pp. 9.11.1–9.11.4.